Interfaz Comparator

- El Comparator está pensado para implementarlo de manera externa a la clase y el Comparable está pensado para ser implementado por el propio objeto.
  Comparable sirve para darle un orden natural a la colección de datos.
- Cuando queremos que exista otras ordenaciones tenedremos que implementar la interfaz comparator.

interfaz **Comparator**

- hay que implementar el método

  compare (Object o1, Object o2)

interfaz **Comparator**

- Por ejemplo, supongamos que queremos ordenar un array de objetos de tipo Persona.

  Una opción sería hacer que Persona implementara Comparable, para después hacer algo como:

  ```
  List<Persona> datos = .....

  datos.add (new Persona(.....));

  Collections.sort (datos);
  ```

  Nótese que la ordenación siempre será la que venga implementada, es decir, ordenará siempre en función de lo definido en el método "compareTo (Object o)".

interfaz **Comparator**

- Ahora bien, si deseas tener varios métodos de ordenación, puedes usar el Comparator implementado en una clase ajena a Persona:

```
public class CompaEdad implements java.util.Comparator {
   …
   public int compare (Object o1, Object o2) {
   ….
   }
   public boolean equals (Object obj) {
   …
   }
}
```

interfaz **Comparator**

- E igualmente puedes hacer muchos más comparadores

```
public class CompaSueldo implements java.util.Comparator {
}
```

- Después, a la hora de ordenar, puedes hacer:

```
Collection.sort (datos, new CompaEdad());
```

o bien

```
Collection.sort (datos, new CompaSueldo());
```